**西 安 邮 电 大 学**

**毕 业 设 计（中期论文）**

题 目：移动端校园活动组织管理系统

学 院： 计算机学院

专 业： 软件工程

班 级： 软件1205班

学生姓名： 王静

导师姓名： 余俊 职称： 工程师

起止时间：2015年11月06日 至 2016年03月14日

西安邮电大学本科毕业设计（论文）开题报告

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 学号 | 04123168 | 姓名 | 王静 | 导师 | 余俊 |
| 题目 | 移动端校园活动组织管理系统 | | | | |
| 选题目的（为什么选该课题）  在日常生活中，大家都比较忙，学习，上班，各种会议等等，这样我们就没有那么多的时间去扩大自己的朋友圈，特别是对于学生来说，如果我们想要组织一个活动一般需要自己主动去联系人，并且还要根据大家的情况去设定时间，地点，并且一般不会限制人数；这样的话就会有很多问题：（1）首先活动的主题，时间，地点都需要发起者一个一个去通知，并且无法保证被联系的人都一定会参加这个活动，比较花费时间；（2）其次就是活动的参与者，一般都是我们自己比较熟悉的人，这样就无法有效的扩大我们的朋友圈。然而随着互联网行业的飞速发展，很多东西都趋向于移动化，每时每刻我们似乎离不开手机，而这也就促成了许多公司的移动化，各大公司都开始专门发展无线端业务以顺应时代的脚步。有了这么便利的条件，基于移动端校园活动组织管理系统就应用而生了，发布者会建立活动主题，时间，地点，以及人数，然后发布活动，分享到自己的朋友圈或者一些群聊中，然后好多人都可以看到，不但可以报名参与，而且也可以分享给其他人，这样既简单便捷，又可以利用熟人关系扩展我们的人脉，更具有一定的实用价值。并且：  1）这种便捷的方式让我们的生活不再孤独，特别是对那些异乡人来说，丰富人们的生活，未来人的生活压力越来越大，这个是个不错的娱乐方式；  2）微信盛行，让很多人变成点赞族，人与人之间的沟通交流越来越少，而移动端这一大趋势刚好可以提高人与人之间的互动，交流，是个不错的选择；  3）只要发布了活动，就可以分享，而且会传播式分享，即使不曾相识也可以参与到其中，既扩大了人们的朋友圈，又会认识很多志同道合的朋友，一起做一些有意义的事情！  4）移动端的发展还只是起步，有很多问题需要解决，比如性能，适配，兼容性…实习的时候接触了移动端，也是热衷于它，也积累了一些经验，被它深深的吸引，未来的职业规划也是移动端，以后就业的岗位也是移动端，所以想多尝试一些移动端的项目，在不断提高自己能力的同时，为人们做一点力所能及的贡献！ | | | | | |
| 前期基础（已学课程、掌握的工具，资料积累、软硬件条件等）  已学课程：计算机导论与C基础、Java语言程序设计A、数据库原理及应用A、数据结构A、算法设计与分析A  掌握的工具：IntelliJ idea、mysql、tomcat、webstorm、chrome调试工具  硬件条件：Windows 操作系统计算机  参考文献：  [1]俞东进，任祖杰 著，《Java EE Web应用开发基础》，电子工业出版社， 2012年  [2]刘乃丽 著，《精通Java EE项目开发——基于Eclipse、Spring、Struts、Hibernate》． 人民邮电出版社，2008年  [3]李杰 柳靖 刘淼 译，《HTML5高级程序设计》，人民邮电出版社 | | | | | |
| [4]陈剑瓯 译，《精通CSS》，人民邮电出版社  [5]大漠 著，《图解CSS3》，机械工业出版社  [6]李松峰 曹力 译，《JavaScript高级程序设计》（第三版），人民邮电出版社  [7]杨涛 王建桥 译，《JavaScript Dom编程艺术》（第二版），人民邮电出版社  [8]淘宝前端团队 译，《JavaScript权威指南》，机械工业出版社  [9]陈娟 译，《HTTP权威指南》，人民邮电出版社  要解决的问题（做什么）  课题主要完成移动端的校园活动组织管理系统，用户可以通过微信授权参与，然后建立活动，发布活动，分享活动；其他人看到这个活动后可以参与其中，也可以分享给别人，也就是利用熟人发起活动  预期目标：   1. 完成一个基于移动端的校园活动组织管理系统，包括前台和后台的基本功能； 2. 利用该系统，用户可以通过微信授权参与，然后建立活动，发布活动，分享活动； 3. 活动包括活动主题，时间，地点，人数； 4. 其他人看到这个活动后可以参与其中，也可以分享给别人 5. 参与者不用注册，只要通过微信授权就可以建立活动，或者参与，分享   具体实现：  1.前端实现  （1）在webstrom中利用html，html5，css，css3搭建基础页面，并用less组织css代码  （2）利用flexible.js做手机屏幕的适配，屏幕适配以及手机端的兼容性问题是难点  （3）用户可以通过点击建立活动按钮，然后填写表单创建活动，最后点击发布，分享按钮  （4）利用javascript实现后台与前端页面的逻辑交互，比如提交表单处理以及验证等  （5）由于是移动端开发，所以性能是最关键的一点，所以前端也要做好各种优化，例如实现模块化，代码压缩，图片压缩，按需加载，缓存等    **2**.后台实现  （1）后台基于java语言开发与微信 API 开发；  （2）利用SpringMVC用于请求分发，利于各个层间的解耦。  （3）数据层使用mysql数据库对数据进行管理。  （4）持久层使用 Mybatis，并使用 Ehcache 作为缓存层，提高效率，可适用于不同数据库。  （5）项目难点在于解决系统并发情况下的访问速度与安全性 | | | | | |
| 工作思路和方案（怎么做）  首先，找一些移动端的开发资料进行研究并及时与老师沟通需求，搞清楚所有需求，接下来写出完整的需求分析文档。然后就是根据需求实现数据库的设计，将每个表之间的数据搞清楚，再在mysql中建立数据库；然后实现框架的搭建，检验每个需求实现代码的健壮性，然后就是进行数据库的链接，实现真正的交互，存储；最终就是UI界面的实际和前端页面的实现，以及前端和后台的交互；  1.前期资料收集。  Web程序开发的框架，以及算法设计相关资料，还有移动端开发资料，以及一些类库。  2.处理收集到的资料，进行分析，进行重点记录。  校园活动组织管理系统的需求分析，及系统框架的设计，数据流向设计，算法选择以及前端类库选择  3.基于分析的数据，对数据库进行设计，并且测试  填写数据，以供开发测试  4.设计并实现所有前端页面，测试多有机型适配以及兼容性问题，以辅助后台逻辑设计，及逻辑视图  5.微信开发，以及编写后台程序代码，将所有页面按照设计的框架及逻辑连接起来，并实现该系统中应该具有的所有功能，以至于整个系统的初步完成。  6.对系统已经初步完成的功能进行组合测试；多找些用户来测试功能，以及压力测试  并完成bug修补。  7.对页面的交互进行测试并记录用户的意见，不断修改，使得页面的交互更加友好。  8.移动端校园活动组织管理系统的最终实现。 | | | | | |
| 指导教师意见  签字 年 月 日 | | | | | |

**西安邮电大学毕业设计 (论文)成绩评**

**目 录**

[摘 要 I](#_Toc28339)

[Abstract II](#_Toc7762)

[1. 系统分析 1](#_Toc22123)

[1.1 系统功能要求 1](#_Toc31987)

[1.2 系统性能要求 1](#_Toc29145)

[2. 系统设计 2](#_Toc6874)

[2.1 功能模块设计 2](#_Toc1321)

[2.2 数据库设计 2](#_Toc24821)

[2.2.1 设计概述 2](#_Toc23899)

[2.2.2 数据库设计图 3](#_Toc22777)

[2.2.3 数据表 3](#_Toc29468)

[2.2.4 数据库E-R图 5](#_Toc8045)

[2.3 系统功能流程图 7](#_Toc1341)

[2.3.1 系统主要流程图 7](#_Toc17080)

[3. 系统功能实现与说明 10](#_Toc12051)

[3.1 创建活动 10](#_Toc23278)

[3.2 修改活动 11](#_Toc27936)

[3.3 参与活动 12](#_Toc3445)

[3.4 分享活动 13](#_Toc5032)

[3.5 活动圈 14](#_Toc18346)

[3.6 活动详情 14](#_Toc9024)

[3.7 个人主页 15](#_Toc1884)

[4. 部分实现代码 20](#_Toc20797)

[4.1 用户授权 20](#_Toc7)

[4.2 响应微信 20](#_Toc19866)

[4.3 活动交互 22](#_Toc6591)

# 1. 系统分析

### 1.1 系统功能要求

本课题主要完成基于移动端的活动组织管理系统，能够按照获取微信的用户信息，分享等接口，并能很好适配各种移动设备。通过该系统，用户可以发布活动，修改活动，分享活动

预期目标：

（1）完成一个B/S体系结构的移动端活动组织系统，包括前台页面显示和后台逻辑处理的基本功能

（2）获取用户的微信账号信息。

（3）获取微信的分享接口。

（4）只要注册微信就可以使用此系统，发布活动，修改活动，参与活动，分享活动等。

（5）解决各种移动终端的适配问题

### 1.2 系统性能要求

移动端活动组织系统为了满足用户的需求应具有以下性能:

（1）系统具有易操作性；

（2）系统具有易维护性；

（3）系统具有可开放性；

（4）系统具用易移植性。

# 2. 系统设计

## 2.1 功能模块设计

移动端活动组织系统主要是微信用户进行操作，主要得业务有发布活动（活动主题，时间，地点，人数）、修改活动(活动主题，时间，地点，人数)，活动详情，活动圈等，如图2-1所示

图2-1 系统功能模块

## 2.2 数据库设计

### 2.2.1 设计概述

为了满足系统的需求和功能的实现，考虑现实的实用性，所以数据库是必不可少，也是前期整个系统构思和设计很大的一部分工作量。根据系统的功能需求分析，由于数据量不是特别大，所以我使用的是mysql数据库。如果按照传统的数据库设计思路，将用户信息直接存入数据库中，虽然会使得逻辑处理变得相当简单，查询效率也会提高，但会造成数据字段大量冗余，同时会浪费数据存储空间，也不利于以后数据方面的更新维护（活动的增加），所以应当设计满足该系统最优的数据存储方式。

### 2.2.2 数据库设计图

活动发布系统管理系统数据库对象主要包括用户信息表、活动信息表、活动参与关联表、用户关联表

### 2.2.3 用户信息表

（1）用户信息表保存了用户的的基本的信息，如用户id、确认令牌、头像链接，昵称，性别等，如表2-1用户信息表所示

用户信息表

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 用户ID | 确认令牌 | 头像链接 | 昵称 | 性别 | 省份 | 城市 | 国籍 | 刷新令牌 | 公众号ID | 刷新时间 |
| userId | accessToken | headImgurl | nickName | sex | Province | City | Country | refreshToken | Openid | refreshTime |

（2）活动信息表保存了活动的基本信息，如活动id、活动类型，活动描述，用户id，活动时间等，如表2-2活动信息表所示。

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 活动ID | 活动类型 | 活动标题 | 活动描述 | 用户ID | 活动开始时间 | 活动创建时间 | 参与人数 | 活动地点 | 已报名人数 | 备注 |
| activityId | typeName | activityTitle | Description | userId | startTime | createTime | personNumber | Location | joinNow | remark |

（3）用户关联表保存了用户的基本的信息，如id、管主用户id、关联热度，关联时间等，如表2-3所示。

表2-3用户关联表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | 主用户ID | 好友ID | 关联热度 | 关联时间 |
| Id | masterId | friendId | Hot | cnTime |

（4）活动参与关联表表保存了活动关联信息，如id，活动id,参与者id，备注，参与状态等，如表2-4所示。

表2-4活动参与关联表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ID | 活动id | 参与者ID | 备注 | 参与状态 | 参与时间 | 参与者昵称 |
| Id | activityId | participator | message | State | joinTime | patName |

### 2.2.4数据库E-R图

1. 用户的E-R图表示了用户的一些属性信息之间的关系。用户信息表的E-R图如图2-3所示。

确认令牌

用户id

省份

性别

昵称

用户

头像链接

公众号id

刷新时间

城市

刷新令牌

国籍

图2-用户信息E-R图

1. 活动信息的E-R图表示了活动信息的一些属性信息之间的关系。活动信息的E-R图如图2-4所示。

活动描述

活动标题

活动类型

活动id

用户id

活动信息

备注

活动开始时间

活动创建时间

已报名人数

参与人数

活动地点

图2-4活动信息E-R图

（3）活动参与关联表的E-R图表示了换的参与关联的一些属性信息之间的关系。它的E-R图如图2-5所示。

活动id

参与时间

参与者昵称

参与者id

活动参与关联表

Id

备注

参与状态

图2-5 活动参与关联表E-R图

（4）用户关联的E-R图表示了用户关联的各个属性信息之间的关系。E-R图如图2-6所示。

备注

备注

备注

用户关联

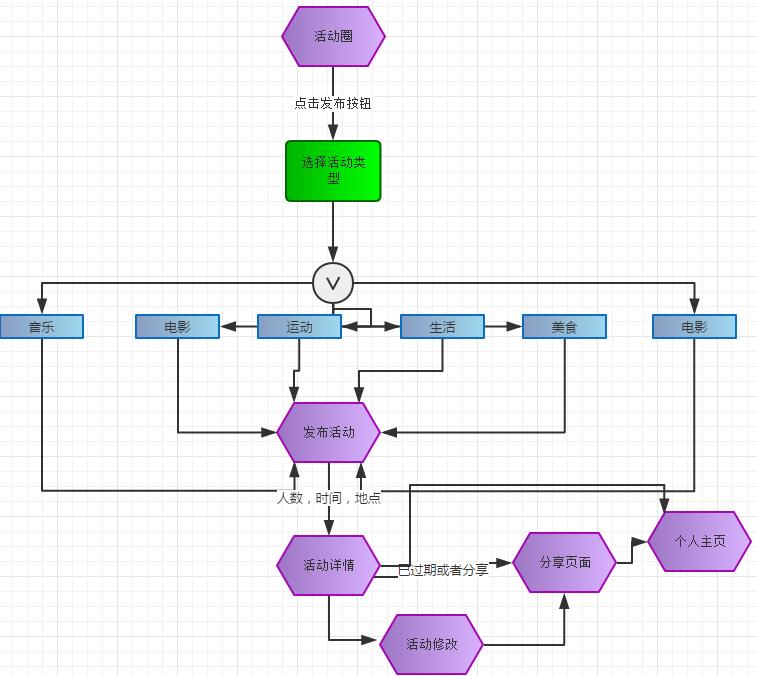
备注

备注

图2-6 用户关联E-R图

## 2.3 系统功能流程图

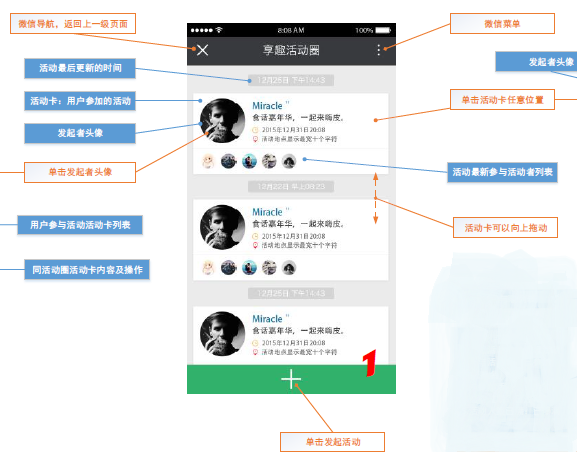
### 2.3.1 系统主要流程图



# 3. 系统功能实现与说明

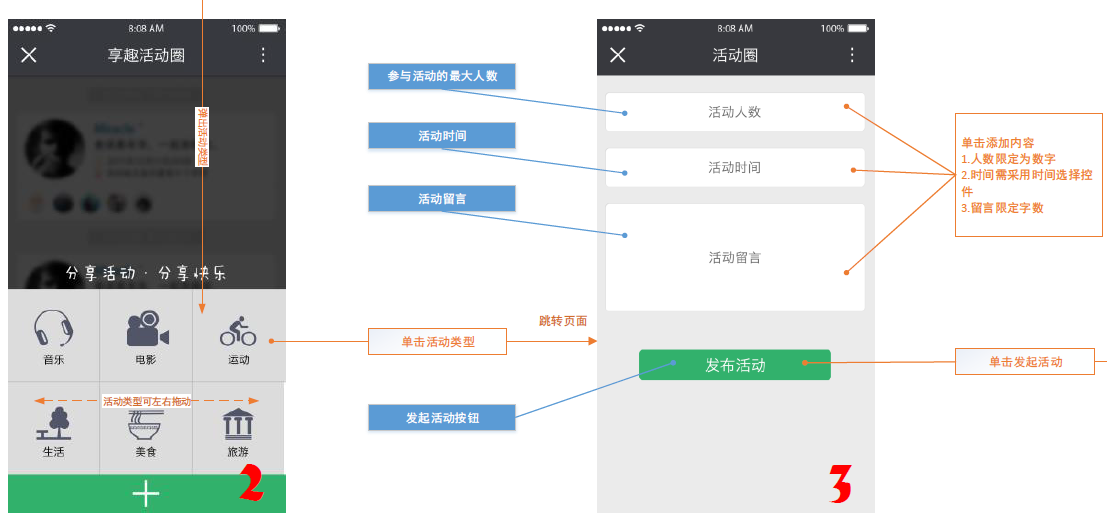
**3.1活动圈：**

用户可以在活动圈看到自己参与或者发起的活动，并可以点击底部的+号按钮，发起活动



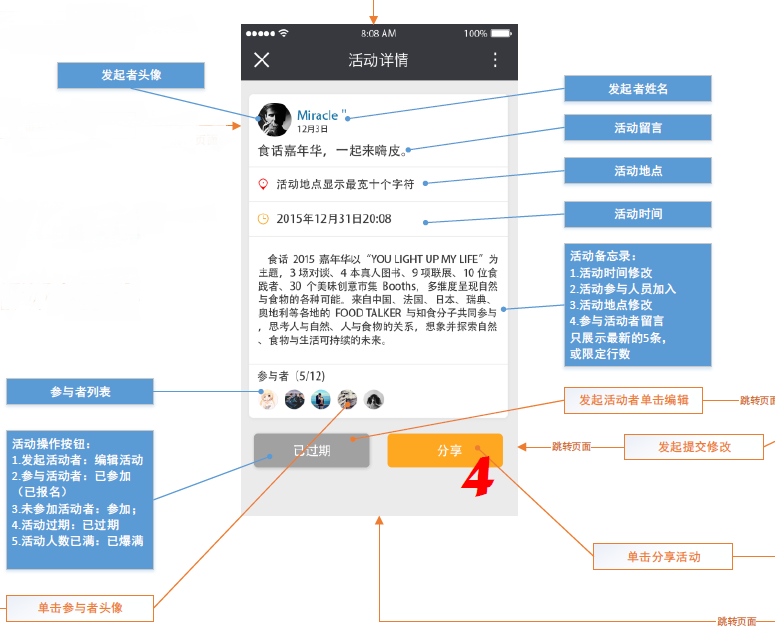
**3.2创建活动：**

用户创建活动时需要选择活动类型，然后在确定活动人数，时间，详情等，提交而发布活动



**3.3活动详情：**

活动详情页面有用户个人信息，活动详细描述，以及参与者列表，分享按钮等



**3.4活动修改：**

活动修改页面，包括时间修改，地点修改，活动说明修改



**3.5分享活动：**

活动提交后，会有一个分享指导，点击后可以选择分享位置



**3.6用户主页：**

用户主页，包含个人信息以及自己发起的活动与参与的活动，可以选择一种分类，查看或者编辑修改等



# 4. 部分实现代码

**[4.1 用户授权：](#_Toc7)**

package com.awareness.trows.controller;

import com.alibaba.fastjson.JSON;

import com.awareness.trows.entity.Account;

import com.awareness.trows.service.AccountService;

import com.awareness.trows.util.wechatSDK.entity.WeChatInfo;

import com.awareness.trows.util.wechatSDK.entity.WxConfig;

import com.awareness.trows.util.wechatSDK.method.AcotMethod;

import com.awareness.trows.util.youbandUtil.CreatID;

import org.apache.commons.codec.digest.DigestUtils;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Controller;

import org.springframework.web.bind.annotation.RequestMapping;

import javax.servlet.ServletException;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import javax.servlet.http.HttpSession;

import java.io.IOException;

import java.io.PrintWriter;

import java.util.HashMap;

import java.util.Map;

import java.util.UUID;

@Controller

/\*\*

\* 与微信对接关键代码

\*

\*/

public class WeChatController {

@Autowired

private AccountService accountService;

/\*\*

\* 状态验证服务 暂时无需开启

\*

\* @param request

\* @param response

\*/

@RequestMapping("/connect")

public void ConnectWechat(HttpServletRequest request, HttpServletResponse response) {

String signature = request.getParameter("signature");

String timestamp = request.getParameter("timestamp");

String nonce = request.getParameter("nonce"); //按照微信要求获取四个参数，前三个供验证是否为微信官方信息

String echostr = request.getParameter("echostr"); //最后为返回信息

//连接测试 忽略验证环节

//========================

try {

PrintWriter out = response.getWriter();

out.print(echostr);

} catch (IOException e) {

e.printStackTrace();

}

}

/\*\*

\* 通过微信API获取用户信息

\*

\* @param request

\*/

@RequestMapping("/getUsrInfo")

public void getUsrInfoByWe(HttpServletRequest request, HttpServletResponse response) {

String code = request.getParameter("code");

String state = request.getParameter("state");

if (null != code && state != null) { //判断用户是否授权 是否有返回code

String requestPath = state.substring(0, state.length() - 1); //请求的页面 表示授权后跳转的页面

String method = state.substring(state.length() - 1); //method为state最后一位o表示静默授权 i表显式授权 r表示刷新 n表示特么的不知道是什么鬼 其余非法

AcotMethod acotMethod = new AcotMethod();

//静默授权处理方式

if (method.equals("o")) { //当授权方式为静默授权时

Account account = accountService.getEntityByStr(acotMethod.GetOpenidByCode(code), "getAccByOid");//获取用户信息

if (account != null) {

int time = (int) (System.currentTimeMillis() - account.getRefreshTime());//计算用户信息的存在时间

if (time > WeChatInfo.overtime) { //如果用户信息过期 让用户使用显式授权state状态为r

String refUrl = "https://open.weixin.qq.com/connect/oauth2/authorize?appid=" + WeChatInfo.appID + "&redirect\_uri=" + WeChatInfo.redirect\_uri + "&response\_type=code&scope=snsapi\_userinfo&state=" + requestPath + "r#wechat\_redirect";

try {

response.sendRedirect(refUrl);

} catch (IOException e) {

e.printStackTrace();

}

} else { //如果用户信息存在且未过期

HttpSession session = request.getSession(true); //获取Session

session.setAttribute("openid", account.getOpenid()); //将用户openid写入seeion

session.setAttribute("nickName", account.getNickName());

session.setAttribute("userId", account.getUserId());

session.setAttribute("headimgurl", account.getHeadimgurl());

try {

response.sendRedirect("." + requestPath); //用户验证成功跳转

} catch (IOException e) {

e.printStackTrace();

}

}

} else { //如果用户不存在 则让用户使用显试授权

try {//state标记为i状态

String InfoUrl = "https://open.weixin.qq.com/connect/oauth2/authorize?appid=" + WeChatInfo.appID + "&redirect\_uri=" + WeChatInfo.redirect\_uri + "&response\_type=code&scope=snsapi\_userinfo&state=" + requestPath + "i#wechat\_redirect";

response.sendRedirect(InfoUrl);

} catch (IOException e) {

e.printStackTrace();

}

}

} else

//显式授权处理方式

{ //当授权方式为显式时

System.out.println("已经调用显示授权方式");

Account account = acotMethod.GetUsrInfoByCode(code);

if (method.equals("i")) {

//表示用户第一次授权 直接写入数据库

account.setUserId(CreatID.getUserID()); //生成用户id

try {

accountService.insert(account, "CreateNewAccount");//把新数据写入数据库

//写入用户头像至图片服务器

acotMethod.PersistenceImg("headimg", account.getUserId() + "", account.getHeadimgurl());

} catch (Exception e) {

System.out.println("========用户信息持久化过程出现异常，已捕获==========");

e.printStackTrace();

System.out.println("首次获取用户信息失败===原因未知 尝试继续完成登入操作");

}

} else

//此时为超期授权

{

if (method.equals("r")) {

accountService.update(account, "updateAcc");//更新数据库

//更新用户头像至图片服务器

acotMethod.PersistenceImg("headimg", account.getUserId() + "", account.getHeadimgurl());

}

}

HttpSession session = request.getSession(true); //获取Session

session.setAttribute("openid", account.getOpenid()); //将用户openid写入seeion

session.setAttribute("nickName", account.getNickName());

session.setAttribute("userId", account.getUserId());

session.setAttribute("headimgurl", account.getHeadimgurl());

try {

response.sendRedirect("." + requestPath); //用户验证成功跳转

} catch (IOException e) {

e.printStackTrace();

}

}

} else { //code为空时 ||授权失败时

try {

if (state != null && state != "") { //判断state是否非法

response.sendRedirect("./activity/error.jsp?requestPath=" + state); //处理用户拒绝授权

}

} catch (IOException e) {

e.printStackTrace();

}

}

}

/\*\*

\* 通过ajax获取微信签名的方法

\*

\* @param request

\* @param response

\*/

@RequestMapping("getSignature")

public void getSignature(HttpServletRequest request, HttpServletResponse response) {

//======================处理微信签名============================

System.out.println("开始签名");

String url = request.getParameter("url");

String timestamp = Long.toString(System.currentTimeMillis() / 1000);

String nonceStr = UUID.randomUUID().toString().substring(0, 30);

String jsapi\_ticket = WxConfig.ticket;

String string1 = "jsapi\_ticket=" + jsapi\_ticket +

"&noncestr=" + nonceStr +

"&timestamp=" + timestamp +

"&url=" + url;

String signature = DigestUtils.sha1Hex(string1);

// System.out.println("==================我是ajax" + signature);

Map<String, String> map = new HashMap<String, String>();

map.put("appId", WeChatInfo.appID);

map.put("nonceStr", nonceStr);

map.put("timestamp", timestamp);

map.put("url", url);

map.put("signature", signature);

map.put("jsapi\_ticket", WxConfig.ticket);

String data = JSON.toJSONString(map);

try {

PrintWriter out = response.getWriter();

out.print(data);

} catch (IOException e) {

e.printStackTrace();

}

// ===========================================================

}

}

**[4.2 响应微信：](#_Toc19866)**

//响应微信API关键代码

var userName = document.getElementById("userName").value;

var desc = document.getElementById("desc").value;

var imgUrl = document.getElementById("img").value;

var link = document.getElementById("basePath").value;

var nowUserId = document.getElementById("nowUserId").value;

var actUserId = document.getElementById("actUserId").value;

var actId = document.getElementById("actId").value;

var shareNow = document.getElementById("shareNow").value + 1;

var pageToken = document.getElementById("pageToken").value;

var ul = document.getElementById("joinInfo");

var joinNow = document.getElementById("joinNow").value;

var perNum = document.getElementById("perNum").value;

var deadLine = document.getElementById("deadLine").value;

var actMember = document.getElementsByClassName("act-member");

window.onload=function(){

$.ajax({

type: 'POST',

url: './getSignature.do',

timeout: 5000,

dataType: 'json', //声明返回文件类型

data: [

{name: "url", value: window.location.href}

],

success: function (data) {

//获取分享所必须的信息

//判断用户属性

if (nowUserId == actUserId) {//当前用户为活动发起者时

link = link + '1/' + actId + '/' + nowUserId + '/getActivityById.do';

} else {

link = link + shareNow + '/' + actId + '/' + nowUserId + '/getActivityById.do';

}

//link 注入完毕

//alert(title+" "+desc+" "+imgUrl+" "+link);

//===================

wx.config({

debug: false,

appId: data.appId,

timestamp: data.timestamp,

nonceStr: data.nonceStr,

signature: data.signature,

jsApiList: [

'checkJsApi',

'onMenuShareTimeline',

'onMenuShareAppMessage',

'hideMenuItems',

'showMenuItems'

]

});

wx.ready(function () {

//当启动时批量隐藏菜单

wx.hideMenuItems({

menuList: [

'menuItem:readMode', // 阅读模式

'menuItem:copyUrl',// 复制链接

'menuItem:share:qq',//分享到qq

'menuItem:share:weiboApp',//分享到微博

'menuItem:share:facebook',

'menuItem:share:QZone',

'menuItem:editTag',//编辑标签

'menuItem:delete',//删除

'menuItem:copyUrl',//复制链接

'menuItem:originPage',//原网页

'menuItem:readMode',//阅读模式

'menuItem:openWithQQBrowser',//在QQ浏览器中打开

'menuItem:openWithSafari',

'menuItem:share:email',

'menuItem:share:brand'

],

success: function (res) {

//alert('成功启动菜单隐藏');

},

fail: function (res) {

//alert(JSON.stringify(res));

}

});

/\*\*

\* 注入分享到朋友的信息

\*/

wx.onMenuShareAppMessage({

title: userName+'发起的活动',

desc: desc,

link: link,

imgUrl: imgUrl + nowUserId + '.jpg',

trigger: function () {

// 不要尝试在trigger中使用ajax异步请求修改本次分享的内容，因为客户端分享操作是一个同步操作，这时候使用ajax的回包会还没有返回

//alert('用户点击发送给朋友');

},

success: function () {

//alert('已分享给朋友');

},

cancel: function () {

alert('已取消');

},

fail: function (res) {

alert(JSON.stringify(res));

}

});

/\*\*

\* 注入分享到朋友圈信息

\*/

wx.onMenuShareTimeline({

title: userName+'发起的活动',

link: link,

imgUrl: imgUrl + nowUserId + '.jpg',

trigger: function () {

// 不要尝试在trigger中使用ajax异步请求修改本次分享的内容，因为客户端分享操作是一个同步操作，这时候使用ajax的回包会还没有返回

//alert('用户点击分享到朋友圈');

},

success: function () {

//alert('已分享至朋友圈');

},

cancel: function () {

alert('已取消');

},

fail: function (res) {

alert(JSON.stringify(res));

}

});

});

}

})

//往下可以继续码需要初始化加载的方法

getJoinInfo();

}

function getJoinInfo(){

$.ajax({

type: 'POST',

url: './getJoinInfo.do',

timeout: 5000,

dataType: 'json', //声明返回文件类型

data: [

{name: "pageNow", value: 1}, //提供所需要的页数

{name: "pageSize", value: 50}, //提供没页的大小

{name: "actId", value: actId} //提供页面的id

],

success: function (data) { //当参数成功返回时

var size = data.length;

var link = "#";

for (var i = 0; i < size; i++) {

var imgLink = imgUrl+data[i].participator+'.jpg';

if(nowUserId==actUserId){

link = './'+data[i].participator+'/getAUserActList.do';

}

addJoinInfo(link,imgLink);

}

}

});

}

function addJoinInfo(link,imgLink){

var li = document.createElement("li");

var htm ='<a href="'+link+'"><img src="'+imgLink+'"></a>';

li.innerHTML = htm;

actMember[0].appendChild(li);

}

**[4.3 活动交互：](#_Toc6591)**

package com.awareness.trows.controller;

import com.alibaba.fastjson.JSON;

import com.awareness.trows.entity.Activity;

import com.awareness.trows.entity.ActPat;

import com.awareness.trows.entity.Relation;

import com.awareness.trows.service.ActPatService;

import com.awareness.trows.service.ActivityService;

import com.awareness.trows.service.RelationService;

import com.awareness.trows.util.youbandUtil.CreatID;

import com.github.pagehelper.PageHelper;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Controller;

import org.springframework.web.bind.annotation.PathVariable;

import org.springframework.web.bind.annotation.RequestMapping;

import javax.servlet.ServletException;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import javax.servlet.http.HttpSession;

import java.io.IOException;

import java.io.PrintWriter;

import java.util.List;

import sun.misc.BASE64Decoder;

/\*\*

\* 活动信息控制器 处理活动信息的curd

\*/

@Controller

public class ActivityController {

@Autowired

private ActivityService activityService; //活动信息持久化服务

@Autowired

private ActPatService actPatService; //活动参与持久服务

@Autowired

private RelationService relationService;

/\*\*

\* 通过页面表单信息添加活动 （持久化）

\*

\* @param request

\* @return

\*/

@RequestMapping("/addActivity")

public String addActivity(HttpServletRequest request) {

HttpSession session = request.getSession(true);

//防止页面后退重复提交

String token = (String) session.getAttribute("token"); //获取当前系统令牌

String pageToken = request.getParameter("pageToken"); //获取页面令牌

//=======================================

if (pageToken.equals(token)) { //令牌验证成功

String actType = null; //活动类型

try {

actType = new String(new BASE64Decoder().decodeBuffer(request.getParameter("actType")),"UTF-8");

} catch (IOException e) {

e.printStackTrace();

}

String activityTitle = request.getParameter("actName"); //活动标题--

String description = request.getParameter("disc"); //活动描述--

String startTime = request.getParameter("appDateTime1"); //开始时间--

String endTime = request.getParameter("appDateTime2"); //结束时间--

String deadLine = request.getParameter("appDateTime3");//报名截止时间--

int personNumber = Integer.parseInt(request.getParameter("actNum"));//活动人数--

String location = request.getParameter("actPlace");//活动地点--

// int shareDepth = Integer.parseInt(request.getParameter("shareDepth"));//分享深度\*\*

long userId = (Long)session.getAttribute("userId");//通过session获取当前用户的id--

String userName = (String) session.getAttribute("nickName");

long activityId = CreatID.getActID(userId); //创建活动id

//=================获取数据的分割线=========================

Activity activity = new Activity(); //创建活动对象

activity.setActivityTitle(activityTitle); //注入活动信息

activity.setActivityId(activityId);

activity.setDescription(description);

activity.setStartTime(startTime);

activity.setEndTime(endTime);

activity.setPersonNumber(personNumber);

activity.setLocation(location);

activity.setShareDepth(1);//&& 控制分享深度

activity.setUserId(userId);

activity.setDeadLine(deadLine);

activity.setTypeName(actType);

activity.setUserName(userName);

activity.setJoinNow(1);

//========================活动注入完毕==========================================

activityService.insert(activity, "addActivity"); //持久化

//新建活动参与对象

ActPat actPat = new ActPat();

actPat.setActivityId(activityId); //注入活动id

actPat.setParticipator(userId); //注入参与者信息

actPat.setPatName(userName);//注入用户名

actPatService.insert(actPat, "joinActivity");//持久化参与信息

//=======================持久化完成 页面跳转处理================================

activity.setUserName(userName); //获取发起者用户名

request.setAttribute("act", activity); //把存好的活动对象放入request

request.setAttribute("shareNow", -1); //向request中存入当前分享层级

session.removeAttribute("token");//提交成功移除token

return "/togmob/actInfo"; //活动发表 持久化成功 带参数跳转至活动详情页面

} else {

request.setAttribute("repeat", true); //token检查出错跳转至欢迎页面

return "/togmob/actCircle";

}

}

/\*\*

\* 通过活动ID获取活动详细信息及已报名人员

\*

\* @param shareNow

\* @param activityId

\* @param request

\*/

@RequestMapping("/{shareNow}/{activityId}/{preUser}/getActivityById")

public String showActivityDetailById(@PathVariable int shareNow, @PathVariable long activityId,@PathVariable long preUser, HttpServletRequest request) {

//获得单个活动的详细信息

Activity activity = activityService.getEntityByKey(activityId, "getActivityById");

if(activity==null) return "/actmob/index"; //如果活动不存在 则跳转至个人主页

HttpSession session = request.getSession(true);

request.setAttribute("act", activity);

request.setAttribute("shareNow", shareNow);

Relation relation = new Relation();

//========持久化用户关系数据===================

long userId = (Long) session.getAttribute("userId");

if (preUser!= userId){

relation.setMasterId(preUser);

relation.setFriendId(userId);

relation.setHot(1);

relationService.insert(relation,"addRelation");

}

//==========================================

return "/shareJoy/activityDetail";

}

/\*\*

\* 通过用户ID获取用户自身发布的活动的列表

\*

\* @param request

\* @return

\*/

@RequestMapping("/{userId}/getAUserActList")

public String getActivityList(@PathVariable long userId,HttpServletRequest request) {

List<Activity> activityList = activityService.getListByKey(userId, "getAUserActList");

request.setAttribute("activityList", activityList);

return "/togmob/myAct";

}

/\*\*

\* 通过用户ID获取用户参与的活动信息列表

\*

\* @param request

\* @return

\*/

@RequestMapping("/getMyJoinActivityListByUserId")

public void getMyJoinActivityListByUserId(HttpServletRequest request, HttpServletResponse response) {

HttpSession session = request.getSession(true);

int pageNow = Integer.parseInt(request.getParameter("pageNow"));

int pageSize = Integer.parseInt(request.getParameter("pageSize"));

long userId = (Long)session.getAttribute("userId");

PageHelper.startPage(pageNow, pageSize); //设定分页信息

List<Activity> activityList = activityService.getListByKey(userId, "getMyJoinActivityListByUserId");//查询数据库

if (activityList != null) {

String jsonStr = JSON.toJSONString(activityList);

System.out.println(jsonStr);

response.setCharacterEncoding("UTF-8");

try {

PrintWriter out = response.getWriter();

out.print(jsonStr); //向客户端返回数据

} catch (IOException e) {

e.printStackTrace();

}

}

}

/\*\*

\* 删除活动（未使用）

\*

\* @param request

\*/

@RequestMapping("delActivity")

public void delActivity(HttpServletRequest request, HttpServletResponse response) {

activityService.deleteByKey(Long.parseLong(request.getParameter("activityId")), "delActivity");

try {

request.getRequestDispatcher("./getActivityListByUserId.do").forward(request, response);

} catch (ServletException e) {

e.printStackTrace();

} catch (IOException e) {

e.printStackTrace();

}

}

/\*\*

\* 修改活动（未完成）

\*

\* @param request

\* @return

\*/

@RequestMapping("alterAct")

public String alterAct(HttpServletRequest request) {

HttpSession session = request.getSession(true);

//防止页面后退重复提交

String token = (String) session.getAttribute("token");

String pageToken = request.getParameter("pageToken");

//=======================================

if (token.equals(pageToken)) {

int activityId = Integer.parseInt(request.getParameter("activityId"));

String activityTitle = request.getParameter("activityTitle");

String description = request.getParameter("description");

String startTime = request.getParameter("startTime");

String endTime = request.getParameter("endTime");

String deadLine = request.getParameter("deadLine");

int personNumber = Integer.parseInt(request.getParameter("personNumber"));

String location = request.getParameter("location");

float privateCost = Float.parseFloat(request.getParameter("privateCost"));

int shareDepth = Integer.parseInt(request.getParameter("shareDepth"));

int joinCheck = Integer.parseInt(request.getParameter("joinCheck"));

String typeName = request.getParameter("typeName");

Activity act = new Activity();

act.setActivityId(activityId);

act.setActivityTitle(activityTitle);

act.setDescription(description);

act.setStartTime(startTime);

act.setEndTime(endTime);

act.setDeadLine(deadLine);

act.setPersonNumber(personNumber);

act.setLocation(location);

act.setPrivateCost(privateCost);

act.setShareDepth(shareDepth);

act.setJoinCheck(joinCheck);

activityService.update(act, "alterAct");

request.setAttribute("activity", act);

session.removeAttribute("token");

return null;

} else {

request.setAttribute("repeat", true);

return "/activity/welcome";

}

}

}